A Variational Technique for Spacecraft Trajectory Planning

Carl Glen Henshaw
Honeywell Technology Solutions, Inc.
ghenshaw@space.nrl.navy.mil
(202) 767-1196

Robert M. Sanner
Space Systems Laboratory
University of Maryland, College Park
rmsanner@eng.umd.edu
(301) 405-1928

August 5, 2003

This paper describes a spacecraft trajectory planning algorithm based on the calculus of variations which can solve 6-DOF spacecraft docking and proximity operations problems. The design of a cost functional which trades off fuel use, obstacle clearance distance, and arrival time is discussed. The nonlinear orbital dynamic equations are treated as dynamic constraints. The Euler–Lagrange equations for this functional are derived, as is the Pontryagin criteria for the optimal control input given realistic saturating on–off thrusters. The indirect collocation method is chosen to solve the attendant boundary–value problem for its lack of sensitivity to initial conditions; continuation is used to further improve the algorithm's robustness. The manipulation of the Euler–Lagrange equations and the transversality condition into a form suitable for use with existing collocation codes is discussed. Trajectories are shown for a variety of orbital maneuvers, culminating in an end–to–end docking maneuver with a tumbling satellite.

1 Introduction

A successful autonomous spacecraft must be able to navigate intelligently. Intelligent navigation entails finding and following collision–free paths through space which can be traversed in a "reasonable" time and which use a "reasonable" amount of fuel. Several spacecraft have been recently proposed for which intelligent navigation abilities are either desirable or necessary, including NASA's Personal Satellite Assistant (PSA) [1], Daimler–Chrysler's ISS Inspector, [2], the European Space Agency's Automated Transfer Vehicle (ATV) [3], and DARPA's Orbital Express vehicle[4]. All of these vehicles must move about in constrained environments; most will also need to contact certain objects in their environment. PSA and Inspector will need to berth themselves to recharge their batteries and fuel supplies; ATV must dock with an ISS docking target; and Orbital Express must grapple satellites.

In order to fulfill the requirements for these robotics missions, trajectory planning algorithms which find low–fuel paths through space while avoiding obstacles must be developed. For ATV and Orbital Express in particular this is a difficult problem since these vehicles will in general start in different orbits from their docking targets; trajectory planners for these vehicles must therefore combine "long range" orbital maneuvering, with its attendent nonlinear dynamics and fuel use problems, and "short range" proximity operations where collision avoidance is paramount.

Techniques for planning trajectories for orbital maneuvering have been used successfully for many years, but these techniques do not generally deal with obstacle avoidance problems. On the other hand, most trajectory planning techniques which do deal with obstacles were developed for terrestrial applications such as wheeled robots or manipulators. Unfortunately, these algorithms do not work well when applied to spacecraft because many of the assumptions upon which they are based on do not hold in space. Notably, dynamic effects cannot be ignored. Also, fuel conservation is typically much more important in space applications than in terrestrial robotic problems. Arrival time may also be an important factor, as for certain orbital maneuvering problems the minimum fuel trajectory may require an infinite travel time. Techniques which can generate paths which satisfy orbital and attitude dynamic constraints, trade off fuel use and obstacle clearance distance, satisfy a broad spectrum of endpoint constraints, deal with moving obstacles, and optimize completion time are therefore needed. This paper describes a family of algorithms based on the calculus of variations which satisfies these requirements. The algorithm utilizes a cost functional which penalizes fuel use and obstacle clearance distance, and finds the trajectory which optimizes this cost functional, subject to a set of boundary constraints which may be specified to allow for fixed or free arrival location and completion time.

2 Background

There are a wide variety of techniques which have been developed for solving specific orbital maneuvering problems. Orbital transfers with free transfer times, such as the Hohmann transfer [15] and the bi–elliptical transfer [14], are well known, as are techniques for intercepting specific points in space with or without specified transfer times [31], [12]. These techniques generally constitute fuel–optimal solutions and respect dynamic constraints. Although they assume impulsive, non–saturating thrusters, this assumption is generally valid given the output ability of chemical thrusters and the time scales of most orbital transfer problems. The techniques do not, however, provide obstacle avoidance, nor do they deal with coupled translation–rotation maneuvers.

There are, additionally, a number of techniques for spacecraft proximity operations in cluttered environments. Cell decomposition techniques typically entail forming a discretized grid–like version of the state space and then searching the resulting discrete space using any of a number of standard search techniques. Cell decomposition has been used to address spacecraft trajectory planning problems, notably by Jackson [16] who used an A*–based search technique to generate 6DOF trajectories for a spacecraft performing proximity operations. Roadmap methods also entail generating an adjacency graph that represents the search space, and then using a standard graph search technique to generate a path from the graph. Unlike cell decomposition methods, however, the graph is not the result of a discrete decomposition of the state space. Instead, the graph attempts to capture the "inherent connectivity" of the space. Voronoi diagrams have achieved some popularity for spacecraft trajectory planning [6], [5]. These approaches deal primarily with the obstacle avoidance criterion. They are not fuel optimal, although the fuel use appears to be reasonable for problems requiring relatively small separation distances. They do not deal with arrival time or endpoint criteria, and do not use realistic thruster models. They also do not explicitly tackle realistic orbital dynamics, nor do they typically plan attitude trajectories.

Artificial potential function guidance is one of the simplest and most popular path planning methods. In its most basic form, a mathematical function called a potential function is constructed which has a global minimum at the goal point and a local maximum centered on each obstacle (which are assumed to be compact) [17], [18]. The path is generated by performing a gradient descent algorithm on the potential function, and projecting the resulting path onto the underlying Euclidean space (figure **??**). Artificial potential function guidance has also achieved some popularity for spacecraft trajectory planning [23], [24], [27]. All of these solutions take into account realistic thrusters, stationary obstacles, and free arrival time. They do not minimize fuel use directly, but because they use Hill's equations and are only intended for "short–range" proximity operations the fuel use is small. On the other hand, they are unsuitable for use in "long–range" orbital maneuvering, do not allow for free ending location, and do not deal well with moving obstacles. None deal with attitude planning, and hence are incapable of solving the coupled attitude/orbital problem needed to dock to a rotating target.

2.1 Variational Techniques

The calculus of variations was developed to solve problems which require finding functions which minimize (or maximize) specified criteria. The calculus of variations problem is [9], [21, p. 179]:

Problem 1 *Given a cost functional* $J[\mathbf{z}(t), \mathbf{u}(t), t] = \int_{t_0}^{t_f} L[\mathbf{z}(t), \mathbf{u}(t), t]\, dt$ *find the trajectory* $\mathbf{z}^*(t)$ *and the control input* $\mathbf{u}^*(t)$ *which minimize (or maximize)* $J[\cdot]$ *and which satisfy the dynamic constraint* $\dot{\mathbf{z}}(t) = \mathbf{f}(\mathbf{z}(t), \mathbf{u}(t), t)$.

Common variations include adding endpoint constraints and static constraints. For trajectory planning problems, terms which penalize fuel use, obstacle clearance distance, trajectory completion time, and so on can be included in the cost functional. The trajectory is generated either by approximating the solution to the cost functional via B–splines or some other approximation technique and solving the resulting optimization problem for the vector of spline weights, or by deriving the Euler–Lagrange differential equations whose solutions minimize the cost functional and solving the associated boundary value problem. Either solution technique can be accomplished symbolically if the cost functional contains only quadratic terms, if the dynamic constraints are linear, and if there are no static constraints. If any of these conditions is not met, however, as is usually the case with spacecraft obstacle avoidance problems, numerical solution techniques must be used. Popular approaches to solving the optimization problem include gradient descent methods, notably sequential quadratic programming (SQP). The shooting method is the most popular method for solving boundary value problems [30, pp. 549–553]. The collocation method is another technique for solving boundary value problems which is gaining in popularity [28].

Variational techniques have been used to solve spacecraft trajectory planning problems in a variety of forms. Simmons, et al [29] use a gradient descent technique to search for impulsive thruster burns which solve six degree–of–freedom docking problems, using quaternions as the attitude parameterization. Simmons' solution minimizes fuel use, takes into account realistic orbital dynamics and thrusters, and allows arrival time to vary. It also avoids obstacles, although the number of obstacles is limited. It does not allow for optimization of arrival location. It's use of Hill's equations makes it unsuitable for "long–range" orbital maneuvering.

Rathinam compares the widely available RIOTS optimization package, which implements the shooting method, and sequential quadratic programming when addressing minimum fuel orbital transfer [25]. Rathinam's interest is in SQP, but in this case the shooting method was superior; he conjectures that this is because "the cost is quadratic in the control forces which are the search variable in RIOTS whereas the cost is very nonlinear in the configuration variables which are the search variables" in his approach. Rathinam does take into account the full nonlinear orbital dynamics and allows free arrival time, but does not address obstacle avoidance or six degree–of–freedom trajectories.

Richards, et al [26] use variational trajectory planning to generate three degree–of–freedom trajectories for a small spacecraft performing proximity operations near the International Space Station. Their cost functional penalizes fuel use and incorporates Hill's equations as a dynamic constraint. Obstacles are treated as static constraints, with an additional state variable added to the state vector for each obstacle. In constrast with normal state variables, which take on continuous values, these variables have different binary values depending on whether or not the spacecraft is in contact with each obstacle. They assume saturating, continuous output thrusters. They then use mixed–integer linear programming [10] to solve the optimization problem, which results in a fast, robust algorithm. They could not, however, generate six degree–of–freedom solutions due to the nonlinear nature of attitude kinematics. The algorithm optimizes arrival time and arrival location. Because they do not use the nonlinear orbital dynamic equations, their approach is unsuitable for "long–range" orbital maneuvering.

3 Mathematical Development

For this paper, a variational technique based on indirect collocation is used. The term "indirect" is used to differentiate between methods which attempt to directly approximate the minimum of the cost functional $J[\cdot]$ in an appropriate function approximate space, such as Sequential Quadratic Programming (the "direct" methods), and those which attempt to solve a set of differential equations whose solutions minimize the cost functional (the "indirect" methods).

Variational trajectory planning allows fuel use and obstacle clearance distance to be traded off, and using an indirect optimization technique allows nonlinear dynamic constraints and on–off, saturating thruster constraints to be satisfied. In addition, both arrival time and arrival location can be optimized. Here, the mathematical underpinnings required to implement an indirect collocation–based 6DOF trajectory planner are summarized.

3.1 Euler–Lagrange Equations

The idea behind indirect variational optimization techniques is to derive a set of differential equations whose solutions minimize $J[\cdot]$. First, augment the cost functional with time–varying Lagrange multipliers in order to enforce the system dynamic constraint:

$$\begin{aligned} J_a[\mathbf{z}(t), \nu(t), t] &= \int_{t_0}^{t_f} L[\mathbf{z}(t), \nu(t), t] \\ &\qquad + \lambda^T(t)\mathbf{f}(\mathbf{z}(t), \nu(t))\ dt \\ &= \int_{t_0}^{t_f} L_a[\mathbf{z}(t), \nu(t), t]\ \ dt \end{aligned}$$

where $\dot{\mathbf{z}}(t) = \mathbf{f}(\mathbf{z}(t), \nu(t), t)$. The vector of Lagrange multipliers $\lambda(t)$ is called the costate vector. It can then be shown [20, pp. 184–188] that solutions of the Euler–Lagrange equations

$$\dot{\mathbf{z}}(t) = \mathbf{f}\left(\mathbf{z}^*(t), \nu^*(t), t\right) \qquad (1)$$

$$\dot{\lambda}(t) = -\frac{\partial L_a}{\partial \mathbf{z}}\left(\mathbf{z}^*(t), \nu^*(t), \lambda(t), t\right) \qquad (2)$$

minimize $J[\cdot]$ subject to the boundary constraints $\mathbf{z}(t_0) = \mathbf{z}_0$, $\mathbf{z}(t_f) = \mathbf{z}_f$ and the dynamic constraint $\dot{\mathbf{z}}(t) = \mathbf{f}(\mathbf{z}(t), \nu(t), t)$ [21, p. 179–180], [20]. Note that the Euler–Lagrange equations are coupled second–order differential equations with split boundary conditions. The Euler–Lagrange equations are generally nonlinear except in the special case where the system dynamics $\mathbf{f}(\cdot)$ are linear and the cost functional contains only quadratic terms.

If the thrusters are unconstrained, the optimal control input $\nu^*(t)$ is chosen to solve [20, p. 187]

$$\mathbf{0} = \frac{\partial L_a}{\partial \nu}(\mathbf{z}(t), \nu(t), t)$$

If the thrusters are saturating or otherwise constrained (such as bang–bang thusters) the optimal control input $\nu^*(t)$ may be chosen according to Pontryagin's minimum principle [20, pp. 227–236]:

$$\begin{aligned} L_a(\mathbf{z}(t), \lambda(t), \nu^*(t), t) \leq L_a(\mathbf{z}(t), \lambda(t), \nu(t), t) \\ \forall \text{ admissible } \nu(t) \end{aligned}$$

where an admissible control input $\nu(t)$ is any input value which satisfies the constraints. In either case it is often possible to solve for $\nu^*(t)$ explicitly in terms of the state and costate:

$$\nu^*(t) = \mathbf{\Upsilon}(\mathbf{z}^*(t), \lambda^*(t)) \,.$$

In order to find a unique solution to the Euler–Lagrange equations, a set of boundary conditions must also be determined. The exact form of the boundary conditions varies depending on the problem at hand, e.g. whether one is interested in fixed or free arrival times and fixed, free, or time dependent starting and arrival locations. The specific boundary conditions are specified according to the requirements of the prolem, such as a known starting location $vecvarz\,(t_0)$; boundary conditions that are less constrained, such as an arrival location which must lie on a surface, are determined according to their constraints and the transversality condition [20, pp. 189–198]. For this paper two cases are of interest. The simplest case, that of a fixed arrival time and fixed arrival location, leads to the boundary conditions

$$\begin{aligned} \mathbf{z}(t_0) &= \mathbf{z}_0 \\ \mathbf{z}(t_f) &= \mathbf{z}_f. \end{aligned}$$

The case of most practical importance for spacecraft docking problems is that of a free arrival time and a time-dependent arrival location $\mathbf{z}_f = \theta\,(\mathbf{t_f})$, which leads to the boundary conditions [20, pp. 190–191]

$$\begin{aligned} L_a(\mathbf{z}(t_f), \nu^*(t_f), \lambda(t_f), t_f) - \lambda^T(t_f)\frac{d\theta}{dt}(t_f) &= 0 \\ \mathbf{z}(t_f) &= \theta(t_f) \\ \mathbf{z}(t_0) &= \mathbf{z}_0. \end{aligned}$$

More care must be taken as the required boundary conditions become more complex. For instance, suppose t_f is fixed and z_f is constrained to lie on a surface described by $\mathbf{m}(\mathbf{z}(t))$ (as is the case with, for instance, a time–constrained orbital transfer problem). In this case, it can be shown [20, pp. 190–191] that the transversality condition can be satisfied if and only if the costate at $t = t_f$ is a linear combination of the gradient vectors $\partial m_1(\cdot)/\partial \mathbf{z}$, $\partial m_2(\cdot)/\partial \mathbf{z}$, ..., $\partial m_k(\cdot)/\partial \mathbf{z}$ of the function $\mathbf{m}(\cdot)$, where k is the dimensionality of $\mathbf{m}(\cdot)$. Therefore, the required boundary conditions are

$$\begin{aligned} \mathbf{m}(\mathbf{z}(t_f)) &= \mathbf{0} \\ -\lambda(t_f) &= \sum_{i=1}^{k} d_i \left[\frac{\partial m_i}{\partial \mathbf{z}}(\mathbf{z}(t_f))\right] \\ \mathbf{z}(t_0) &= \mathbf{z}_0. \end{aligned}$$

where d_1, d_2, ..., d_k are constants which must be found simultaneously. If t_f is also free, then the boundary conditions are

$$\begin{aligned} L_a(\mathbf{z}(t_f), \nu^*(t_f), \lambda(t_f), t_f) &= 0 \\ \mathbf{m}(\mathbf{z}(t_f)) &= \mathbf{0} \\ -\lambda(t_f) &= \sum_{i=1}^{k} d_i \left[\frac{\partial m_i}{\partial \mathbf{z}}(\mathbf{z}(t_f))\right] \\ \mathbf{z}(t_0) &= \mathbf{z}_0. \end{aligned}$$

3.2 Collocation

In order to solve the resulting boundary value problem, a variety of techniques may be used. The most popular of these is the shooting method and its variants [30, pp. 549 – 553]. The shooting method entails determining an initial estimate for the unknown part of the initial conditions, integrating the differential equations from t_0 to t_f, and using the error between the desired and actual final conditions to update the estimate using Newton's method or a related method. Shooting is considered to be quite accurate, but is sensitive to the accuracy of the estimate of the initial conditions. Unfortunately, there is often no obvious physical significance to the costate equations, and therefore determining an accurate initial estimate may be difficult.

The collocation method, although not generally as accurate as the shooting method, is less sensitive to the accuracy of the initial solution estimate. The collocation method entails approximating the solution to the differential equations using a cubic spline or other approximation technique [30, p. 548], [28], [19], [8]:

$$\begin{aligned} \mathbf{z}(t) &\approx \mathbf{c}_z^T \mathbf{b}(t) \qquad (3) \\ \lambda(t) &\approx \mathbf{c}_\lambda^T \mathbf{b}(t). \qquad (4) \end{aligned}$$

Here, $\mathbf{c}_z$ and $\mathbf{c}_\lambda$ are vectors of weights and $\mathbf{b}(t)$ is a vector of basis functions. Each basis function is centered on a "knot point" $t_0, t_1, \dots, t_{m-1}$, where the series of knots points discretizes time. The collocation method chooses weights $\mathbf{c}_z$ and $\mathbf{c}_\lambda$ so that the interpolated solution satisfies the boundary conditions and satisfies ("collocates") the differential equations at each of the knot points:

$$\begin{aligned} \mathbf{c}_z^T \dot{\mathbf{b}}(t_i) &= \mathbf{f}\left(\mathbf{c}_z^T \mathbf{b}(t_i), \boldsymbol{\Upsilon}\left(\mathbf{c}_z^T \mathbf{b}(t_i), \mathbf{c}_\lambda^T \mathbf{b}(t_i), t_i\right), t_i\right) \\ \mathbf{c}_\lambda^T \dot{\mathbf{b}}(t_i) &= -\frac{\partial J}{\partial \mathbf{z}}\left(\mathbf{c}_z^T \mathbf{b}(t_i), \mathbf{c}_\lambda^T \mathbf{b}(t_i),\right. \\ &\qquad \left.\boldsymbol{\Upsilon}\left(\mathbf{c}_z^T \mathbf{b}(t_i), \mathbf{c}_\lambda^T \mathbf{b}(t_i), t_i\right), t_i\right) \\ \mathbf{c}_z^T \mathbf{b}(t_0) &= \mathbf{z}_0 \\ \mathbf{c}_z^T \mathbf{b}(t_{m-1}) &= \mathbf{z}_f \end{aligned}$$

where the first two equations come from the collocation requirement and the last two ensure that the boundary conditions are satisfied. The boundary–value problem is therefore transformed into an algebraic matrix equation (AME) which must be solved for the weights $\mathbf{c}_z$ and $\mathbf{c}_\lambda$. This results in $4n \cdot m$ algebraic equations, where n is the size of the system state space and m is the number of the knots:

$$\mathbf{0} = \boldsymbol{\Psi}(\mathbf{c}_z, \mathbf{c}_\lambda) = \boldsymbol{\Psi}(\xi)$$

which must be solved for the weights ξ. To perform this transformation it is assumed that the right–hand side of the differential equations are Lipshitz. For this research, the AME was solved using the Levenberg–Marquardt method, which attempts to combine the robustness of gradient descent with the rapid convergence rate of Newton's method.

In order to strengthen the robustness of the collocation method even further, the AME solver may be combined with a continuation algorithm, which is often used to improve the convergence of collocation schemes for highly nonlinear problems [8], [28], [19]. Continuation modifies the equation to be solved, for instance, by multiplying nonlinear terms in the differential equations by a gain. The gain is first set to zero, leading to a linear problem, which can easily be solved. At each iteration the gain is slowly increased and the solution from the previous step is used as the initial estimate for the current step. When the gain reaches unity the solution to the original problem is found.

3.3 Attitude Representation

In order to plan 6DOF trajectories, the system state space must be augmented with a representation of attitude. All minimal (three element) attitude representations are known to have dynamic singularities somewhere in their range. This fact accounts in part for the popularity of non–minimal representations such as the quaternion, which does not possess any dynamic singularities. Unfortunately, for trajectory planning problems, the use of non–minimal representations causes a serious problem: the Jacobian of the collocation AME is singular, which presents problems for a gradient–based AME solver such as Levenberg–Marquardt. One way to overcome these problems is to use a minimal attitude representation with dynamic singularities in benign places. One such representation is the modified Rodrigues vector, a relatively new three parameter representation which is formulated so that the only singularity occurs at rotations of 360° from the origin

[32]. The elements $\sigma = [\sigma_1, \sigma_2, \sigma_3]^T$ of the modified Rodrigues vector are defined in terms of the quaternion elements $\mathbf{q} = \left[\eta, \epsilon^T\right]$ as

$$\sigma = \frac{\epsilon}{1+\eta}$$

and, conversely,

$$\begin{aligned} \epsilon &= \frac{2\sigma}{1+\sigma^T\sigma} \\ \eta &= \frac{1-\sigma^T\sigma}{1+\sigma^T\sigma}. \end{aligned}$$

The modified Rodrigues vector can be related to the Euler axis $\mathbf{k}$ and angle Φ as

$$\sigma = \mathbf{k} \tan\left(\frac{\Phi}{4}\right)$$

which is well defined for Euler angles in the range $0 \leq \Phi < 360°$. The dynamic equation for the modified Rodrigues vector formulation is

$$\dot{\sigma} = \mathbf{G}_\sigma(\sigma)\,\omega = \frac{1}{2}\left(\mathbf{I} - \mathbf{S}(\sigma) + \sigma\sigma^T - \frac{1+\sigma^T\sigma}{2}\mathbf{I}\right)\omega. \tag{5}$$

3.4 Dynamic Equations

For numerical reasons, here the orbital dynamics will be expressed with respect to the position of the docking target. Combining the two–body orbital dynamics with the modified Rodriguez vector attitude representation yields the 6DOF dynamic equations

$$\begin{aligned} \dot{\mathbf{z}}(t) = \begin{bmatrix} \dot{\tilde{\mathbf{v}}} \\ \dot{\tilde{\mathbf{p}}} \\ \dot{\omega} \\ \dot{\sigma} \end{bmatrix} &= \begin{bmatrix} -\mu\mathbf{p}/\|\mathbf{p}\|^3 - \mathbf{a}_t(t) \\ \tilde{\mathbf{v}} \\ -\mathbf{H}^{-1}\mathbf{S}(\mathbf{H}\omega)\omega \\ \mathbf{G}_\sigma(\sigma)\,\omega \end{bmatrix} \\ &+ \begin{bmatrix} \mathbf{R}(\sigma)\mathbf{u}(t)/m \\ \mathbf{0} \\ \mathbf{H}^{-1}\tau(t) \\ \mathbf{0} \end{bmatrix} \end{aligned} \tag{6}$$

where $\tilde{\mathbf{p}}$ represents the position of the spacecraft in a nonrotating frame whose origin is coincident with the docking target, $\tilde{\mathbf{v}}$ is the associated translational velocity vector, $\mathbf{a}_t(t)$ is the acceleration of the docking target, σ is the modified Rodrigues vector, and ω is the rotational rate vector expressed in the body frame. Also, m is the mass of the spacecraft and $\mathbf{H}$ is the rotational inertia vector. The matrix $\mathbf{R}(\sigma)$ is the rotation matrix that converts vectors in body coordinates to inertial coordinates. This matrix is most conveniently expanded in terms of the equivalent quaternion as

$$\mathbf{R}\left(\mathbf{q}\right) = \left(\eta^2 - \epsilon^T\epsilon\right)\mathbf{I} + 2\epsilon\epsilon^T - 2\eta\,\mathbf{S}\left(\epsilon\right).$$

4 Six Degree–of–Freedom Orbital Maneuvering

To illustrate the application of these ideas to spacecraft trajectory planning, an example docking problem will be presented. For this problem, the spacecraft will begin at a fixed position and attitude, and will be required to dock to a stationary target in the presence of environmental obstacles. Both the docking position and attitude wil be specified; hence the problem is a 6DOF trajectory planning problem, with the translation and attitude dynamics coupled in general. The example will assume that no gravitational bodies are present; a fixed arrival time is also assumed.

The cost functional which will be used for this example has the form

$$J[\cdot] \quad = \quad \int_{t_0}^{t_f} \left(L_{\text{control}}[\mathbf{z}, \nu, \tau] + L_{\text{obstacle}}[\mathbf{z}] + L_{\text{time}} + \lambda^T\mathbf{f}(\mathbf{z}, \nu, \tau)\right) dt$$

$$\begin{aligned} = \quad & \int_{t_0}^{t_f} \big(\|\mathbf{u}(t)\|_1 + \tau(t)^T \mathbf{R} \tau(t) \\ & \qquad + \alpha \sum_{i=1}^{n} g_i(\mathbf{v})\, h_i(\mathbf{p}) + \gamma + \lambda^t \mathbf{f}(\mathbf{z}, \mathbf{\Psi}(\mathbf{z}, \lambda)) \Big)\, dt \end{aligned} \tag{7}$$

where $\mathbf{f}(\cdot)$ represents the spacecraft dynamic equations (7), $L_{\text{control}}[\cdot] = \|\mathbf{u}(t)\|_1 + \tau(t)^T \mathbf{W} \tau(t)$ penalizes control effort, $L_{\text{obstacle}}[\cdot] = \alpha \sum_{i=1}^{n} g_i(\mathbf{v})\, h_i(\mathbf{p})$ penalizes obstacle clearance distance, and $L_{\text{time}} = \gamma$ penalizes completion time.

4.1 Fuel Use

A cost functional of the form

$$J[\cdot] = \int_{t_0}^{t_f} \|\mathbf{u}(t)\|_1 + \cdots \tag{8}$$

results in minimum fuel trajectories [20, pp. 259–284]. However, the control law which results from this cost functional,

$$u_i^*(t) = \begin{cases} u_{max} & \text{if } \lambda_{v,i} < -1/m \\ 0 & \text{if } -1/m \le \lambda_{v,i} \le 1/m \\ -u_{max} & \text{if } \lambda_{v,i} > 1/m \end{cases} \tag{9}$$

is discontinuous, and therefore leads to Euler–Lagrange equations whose right–hand–side is also discontinuous. To overcome this problem, a continuous control law which approximates equation 9 is used (see figure 1):

$$\begin{aligned} s_1 &= \frac{m u_{max}(\epsilon - 1)}{2(e_{max}^2 - 1)\epsilon + 2m^2 u_{max}} \\ s_2 &= \frac{(u_{max} m^2 - 1)\epsilon + u_{max} m^2 - 1}{2m((u_{max} m^2 + 1)\epsilon + u_{max} m^2 - 1)} \\ i_1 &= \frac{(2u_{max}^2 m^2 - u_{max})\epsilon}{(u_{max} m^2 - 1)\epsilon + u_{max} m^2} \\ i_2 &= \frac{(u_{max} m^2 - 1)\epsilon^2 - (3u_{max} m^2 - 1)\epsilon}{2m^2((u_{max} m^2 + 1)\epsilon + u_{max} m^2 - 1)} \\ u_i(t) &= \begin{cases} u_{max} & \text{if } \lambda_{v,i} < -2m \cdot u_{max} \\ s_1 \lambda_{v,i} + i_1 & \text{if } -2m \cdot u_{max} \le \lambda_{v,i} \le -((1-\epsilon)m \cdot u_{max} + \epsilon/m) \\ s_2 \lambda_{v,i} + i_2 & \text{if } -((1-\epsilon)m \cdot u_{max} + \epsilon/m) \le \lambda_{v,i} \le -1/m \\ -\frac{1-\epsilon}{2m^2} m \lambda_{v,i} & \text{if } -1/m \le \lambda_{v,i} \le 1/m \\ s_1 \lambda_{v,i} - i_2 & \text{if } 1/m \le \lambda_{v,i} \le ((1-\epsilon)m \cdot u_{max} + \epsilon/m) \\ s_2 \lambda_{v,i} - i_1 & \text{if } ((1-\epsilon)m \cdot u_{max} + \epsilon/m) \le \lambda_{v,i} \le 2m \cdot u_{max} \\ -u_{max} & \text{if } \lambda_{v,i} > 2m \cdot u_{max}. \end{cases} \end{aligned} \tag{10}$$

where the control law exactly equals the minimum fuel law when the parameter ϵ equals 1. In most of the example in this paper, the continuation method was used starting with $\epsilon = 0$ and progressed until $\epsilon \approx 1$.

A minimum energy cost functional for the rotational actuators will be used:

$$J[\cdot] = \int_{t_0}^{t_f} \cdots + \rho\, \tau(t)^T \tau(t) + \cdots$$

where ρ is a positive constant. The resulting control law is

$$\tau_i^*(t) = \begin{cases} \tau_{max} & \text{if } (\lambda_v^T \mathbf{H}^{-1})_i < -2\rho \cdot \tau_{max} \\ -(\lambda_v \mathbf{H}^{-1})_i / 2\rho & \text{if } -2\rho \cdot \tau_{max} \le (\lambda_v^T \mathbf{H}^{-1})_1 \le 2\rho \cdot \tau_{max} \\ -u_{max} & \text{if } (\lambda_v^T \mathbf{H}^{-1})_1 > 2\rho \cdot \tau_{max}. \end{cases} \tag{11}$$

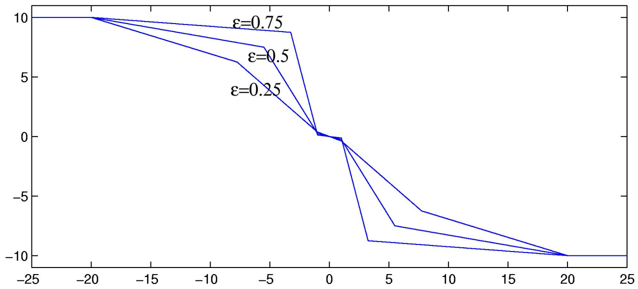


Figure 1: Control law 10 for $u_{max} = 10$, $m = 1$.

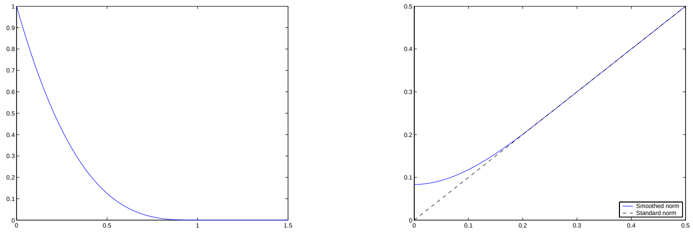


Figure 2: Piecewise cubic obstacle penalty function.; “Smoothed” velocity normalization function.

4.2 Obstacle Clearance

In addition to a term penalizing fuel use, a term penalizing distance to obstacles is also added to the cost functional:

$$J(\cdot) = \int_{t_0}^{t_f} \cdots + \sum_{j=1}^{n} \alpha_j g\big(d_r(\mathbf{p}, \mathbf{p}_{O_j}, \sigma_{O_j})\big) + \cdots \tag{12}$$

where $\mathbf{p}_{O_j}$ denotes the position of the origin of a coordinate frame affixed to the jth obstacle, O_j, and σ_{O_j} represents the orientation of the obstacle coordinate frame. The function $d_r(\cdot)$ is a distance metric from the spacecraft to the obstacle, and is a function of the obstacle shape; α_j is a constant scalar gain. The obstacle penalty function $g(\cdot)$ takes on larger values as the distance to the obstacle boundary decreases, and becomes very small or zero at some distance from the obstacle.

In order for the Euler–Lagrange equations to fulfill the Lipschitz smoothness condition, $\partial g(d_r)/\partial d_r$ must be Lipschitz; this is true if, for instance, $g(\cdot)$ is C^2. One such function consists of a piecewise cubic:

$$\begin{aligned} d_n &= d_r(\mathbf{p}, \mathbf{p}_{O_j}, \sigma_{O_j}) \cdot s \\ g(d_n) &= \begin{cases} k \cdot \left(a_1 d_n^3 + a_2 d_n^2 + a_3 d_n + a_4\right) & d_n < 1 \\ 0 & \text{else.} \end{cases} \end{aligned}$$

where s determines the width of the support interval (the region where $g(\cdot)$ is nonzero) by scaling the relative distance d_r, k determines the maximum height of the penalty function, and the coefficients $a_1, ..., a_4$ are chosen such that $g(0) = k$, $g(s) = 0$, $(\partial g(d_r)/\partial d_r)(s) = 0$, and $\left(\partial^2 g(d_r)/\partial d_r^2\right)(s) = 0$; see figure 2.

Notice that there are two strategies which could be used to minimize 12. One is to reduce the obstacle clearance distance by moving the trajectory away from obstacles. The other is to reduce the time spent close to obstacles by increasing velocity in their vicinity (in fact, the cost can be made as low as desired simply by allowing $t_f \rightarrow t_0$). Clearly, the second approach is undesirable. A term which compensates for increased velocity relative to the obstacles is therefore needed:

$$J(\cdot) = \int_{t_0}^{t_f} \cdots + \sum_{j=1}^{n} \alpha_j g\big(d_r(\mathbf{p}, \mathbf{p}_{O_j}, \sigma_{O_j})\big)\, h\big(v_r(\mathbf{p}, \mathbf{p}_{O_j}, \sigma_{O_j}, \mathbf{v}, \mathbf{v}_{O_j}, \omega_{O_j})\big) + \cdots \tag{13}$$

where $\mathbf{v}_{O_j}$ is the velocity of the jth obstacle’s coordinate frame and ω_{O_j} is the rotational rate of the frame. The relative velocity v_r between the spacecraft and an obstacle is easily defined as the time derivative of the relative distance, and the velocity normalization function $h(\cdot)$ is a function which increases as v_r gets larger.

It is easy to show that in order to exactly correct for changes in velocity, $h(v_r)$ should be $|v_r|$; unfortunately, though, the partial derivative of the absolute value is not defined at $v_r = 0$, which means the Euler–Lagrange equations are not defined at $v_r = 0$ either. To alleviate this problem, the absolute value function is "smoothed" using a patched cubic near $v_r = 0$:

$$h(v_r) = \begin{cases} |v_r| & \text{if } |v_r| > \varepsilon \\ b_1|v_r|^3 + b_2|v_r|^2 + b_3|v_r| + b_4 & \text{else.} \end{cases}$$

where the coefficients $b_1, ..., b_4$ are chosen so that $h(\varepsilon) = |\varepsilon|$, $(\partial h(v_r)/\partial v_r)(\varepsilon) = 1$, $\left(\partial^2 h(v_r)/\partial v_r\right)(\varepsilon) = 0$, and $(\partial h(v_r)/\partial v_r)(0) = 0$ (see figure 2). Because $\partial^2 h(v_r)/\partial v_r^2$ exists and is bounded, this function also satisfies the Lipschitz condition.

4.3 Euler-Lagrange Equations

The Euler–Lagrange equations corresponding to this cost functional are

$$\begin{aligned}
\dot{\mathbf{p}} &= \mathbf{v} \\
\dot{\mathbf{v}} &= -\frac{\mu}{\|\mathbf{p}\|^3}\mathbf{p} + \mathbf{R}(\sigma)\mathbf{u}(t) \\
\dot{\lambda}_p &= \left(\frac{\mu}{\|\mathbf{p}\|^3}\mathbf{I} - \frac{3\mu}{\|\mathbf{p}\|^5}\mathbf{p}\mathbf{p}^t\right)\lambda_v \\
&\quad + \sum_{j=1}^{n} \alpha_j \left(-h(v_{r,j})\,\frac{\partial g(d_{r,j})}{\partial d_{r,j}}\,\frac{\partial d_{r,j}}{\partial \mathbf{p}} - \frac{\partial h\,(v_r)}{\partial v_r}\,\frac{\partial v_{r,j}}{\partial \mathbf{p}} g(d_{r,j})\right) \\
\dot{\lambda}_v &= -\lambda_p + \sum_{j=1}^{n} -\alpha_j \frac{\partial h(v_{r,j})}{\partial v_{r,j}}\,\frac{\partial v_{r,j}}{\partial \mathbf{v}} g(d_{r,j}) \\
\dot{\sigma} &= \mathbf{G}_\sigma(\sigma)\,\omega \\
\dot{\omega} &= \mathbf{H}^{-1}\left(-\mathbf{S}(\mathbf{H}\omega)\,\omega + \tau(t)\right) \\
\dot{\lambda}_\sigma &= -\left(\frac{\partial \mathbf{G}_\sigma(\sigma)}{\partial \sigma}\omega\right)^T \lambda_\sigma \\
\dot{\lambda}_\omega &= \left(\frac{\partial \mathbf{S}(\mathbf{H}\omega)\,\omega}{\partial \omega}\right)^T \mathbf{H}^{-1,T}\lambda_\omega - \mathbf{G}_\sigma(\sigma)\,\lambda_\sigma
\end{aligned} \tag{14}$$

with $\mathbf{u}(t)$ computed as in equation 10 and $\tau(t)$ as in equation 11. The quantities g, h, and d are as defined above.

5 Results

5.1 Optimal Orbital Transfers

The first example attempts to recreate the well–known solution for minimum–fuel two–impulse orbital transfers between coplanar circular orbits, the Hohmann transfer. Hohmann transfers assume that the spacecraft thrusters are impulsive, i.e. that they are capable of infinite output and therefore can impart instantaneous changes in velocity. The Hohmann transfer entails an initial burn which is parallel to the velocity vector of the spacecraft, and of a magnitude that places the spacecraft on an elliptical transfer orbit with a periapse equal to the radius of the initial circular orbit (in this case, 1 A.U.), and an apoapse equal to the radius of the second orbit (here, 2 A.U.). When the spacecraft has traversed a transfer angle of $180°$, and hence reached apoapse, the orbit is circularized with a burn that is, again, parallel to the velocity vector of the spacecraft. The total delta-V for this transfer is thus ≈ 1.7873 A.U./year. If the spacecraft has a mass of 1 unit and that the spacecraft thrusters have a very high output then the total fuel required to carry out the Hohmann transfer is ≈ 1.7873 units. The transfer time is ≈ 0.9186 years.

In order to solve this problem using the variational techniques outlined in this paper, a cost functional must first be specified. The cost functional

$$J[\cdot] = \int_{t_0}^{t_f} \|\mathbf{u}(t)\|_1 \; dt. \tag{15}$$

results in minimum–fuel trajectories assuming the thrusters are aligned with the body frame and do not rotate [20, p. 261]. Applying equations 1 and 2 to this cost functional yields the Euler–Lagrange equations

$$\begin{bmatrix}\dot{\mathbf{v}}\\ \dot{\mathbf{p}}\\ \dot{\lambda}_v\\ \dot{\lambda}_p\end{bmatrix} = \begin{bmatrix} -\mu\mathbf{p}/\|\mathbf{p}\|^3\\ \mathbf{v}\\ -\left(\frac{3\mu\mathbf{p}\mathbf{p}^T}{\|\mathbf{p}\|^5} - \frac{\mu}{\|\mathbf{p}\|^3}\mathbf{I}\right)\lambda_p\\ -\lambda_v\end{bmatrix} + \begin{bmatrix}\mathbf{u}(t)/m\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix}. \tag{16}$$

The control law (10) was used here for $\mathbf{u}(t)$.

For this type of problem the arrival time is free. Since both the initial and target orbits are circular, the initial position and velocity and the start time may be assumed to be fixed without loss of generality. The final position and velocity are required to lie on the target orbit. In order to determine the required boundary conditions for this problem a function $\mathbf{m}(\cdot)$ where $\mathbf{m}(\mathbf{z}(t)) = \mathbf{0}$ on the target orbit must be defined. One such function is

$$\mathbf{m}(\mathbf{p},\mathbf{v}) = \begin{bmatrix} p_3\\ v_3\\ \mathbf{p}\cdot\mathbf{v}\\ \|\mathbf{p}\| - r_t\\ \|\mathbf{v}\| - v_t\end{bmatrix}$$

where the first two terms ensure that the orbit is in the x–y plane, the third ensures that the velocity vector is orthogonal to the position vector (which is always the case for a circular orbit), and the fourth and fifth ensure that the orbit has the required radius and velocity, which are constant for a circular orbit. For the currect example, $r_t = 2$ and $v_t = \sqrt{2}\pi$. The boundary conditions are then found to be

$$\mathbf{p}(t_0) = \begin{bmatrix}0\\ r_0\\ 0\end{bmatrix} \tag{17}$$

$$\mathbf{v}(t_0) = \begin{bmatrix}v_0\\ 0\\ 0\end{bmatrix} \tag{18}$$

$$\|\mathbf{u}(t_f)\|_1 + \lambda^T(t_f)\mathbf{f}(\mathbf{p}(t_f),\mathbf{v}(t_f),\mathbf{u}(t_f)) = 0 \tag{19}$$

$$\mathbf{m}(\mathbf{p}(t_f),\mathbf{v}(t_f)) = \mathbf{0} \tag{20}$$

$$\begin{aligned} -\begin{bmatrix}\lambda_p(t_f)\\ \lambda_v(t_f)\end{bmatrix} = d_1\begin{bmatrix}0\\0\\1\\0\\0\\0\end{bmatrix} + d_2\begin{bmatrix}0\\0\\0\\0\\0\\1\end{bmatrix} + d_3\begin{bmatrix}v_1(t_f)\\ v_2(t_f)\\ v_3(t_f)\\ p_1(t_f)\\ p_2(t_f)\\ p_3(t_f)\end{bmatrix}\\ +d_4\begin{bmatrix}p_1(t_f)/\|\mathbf{p}(t_f)\|\\ p_2(t_f)/\|\mathbf{p}(t_f)\|\\ p_3(t_f)/\|\mathbf{p}(t_f)\|\\ 0\\ 0\\ 0\end{bmatrix} + d_5\begin{bmatrix}0\\ 0\\ 0\\ v_1(t_f)/\|\mathbf{v}(t_f)\|\\ v_2(t_f)/\|\mathbf{v}(t_f)\|\\ v_3(t_f)/\|\mathbf{v}(t_f)\|\end{bmatrix}. \end{aligned} \tag{21}$$

where, here, $r_0 = 1$ and $v_0 = 2\pi$.

The Euler–Lagrange equations 16 were solved subject to the boundary conditions 20 – 21. The initial solution estimate was generated by fitting a cubic polynomial to the initial and final arrival points (which were assumed to be the known optimal points). The optimal arrival time was initially estimated to be $t_f = 0.9168$ years. The mesh had ten nodes uniformly distributed on $[0, 1]$. The resulting trajectory is shown in figure 3. The thrusters were assumed to saturate at 500,000 thrust units (a large number was chosen to approximate the impulsive thrust assumed by a Hohmann transfer),

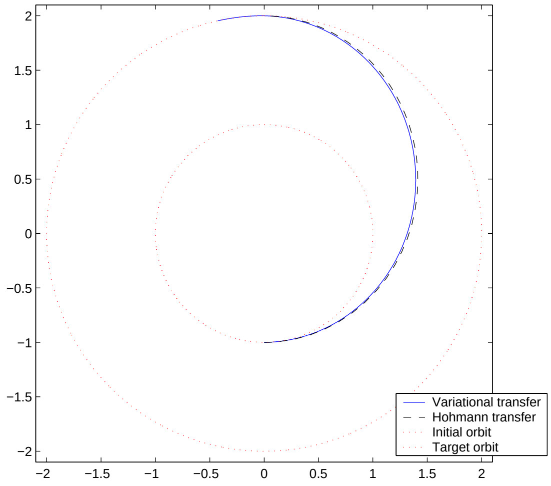


Figure 3: Variational transfer

and the spacecraft mass was 1 unit. Note that the trajectory found by the variational technique qualitatively resembles a Hohmann transfer. The time of arrival was 1.008 years, slightly longer than the 0.9186 years required by the Hohmann transfer. The total fuel required for the variational trajectory is 1.7961 units, only 0.5% higher than the Hohmann transfer.

It should be noted that this problem was solved using continuation, starting with the control law 10 parameter ϵ set at zero. As mentioned, as ϵ approaches unity, control law 10 approaches the true minimum–fuel control law, and the variational solution approaches the true Hohmann transfer trajectory. For this problem, the final solution given is for $\epsilon = 0.999$. The fact that ϵ cannot be exactly one because this would violate the Lipschitz assumption, as discussed above, explains the small variation between the variational solution and a true Hohmann transfer trajectory.

5.2 Free–space Docking

This example requires the vehicle to start with zero velocity at a fixed point in space and dock with a stationary, spherical target. The target has a radius of 0.5 units and is centered at (2.4, 3.0, 0.0). The required endoint is located at (3.0, 3.0, 0.0) with a closing velocity of (-0.5, 0.0, 0.0) units per second. The final attitude is required to be ”nose–in”, i.e. with the x–axis of the body frame tangent to the surface of the target. Note that the endpoint does not lie exactly on the target sphere. Ordinarily, the final part of the trajectory for a docking maneuver will be predetermined, and there is no planning to be done. This allows the user to design a final approach which satisfies highly complex mechanical mating constraints and safety criteria. Here it is assumed that the predetermined part of the docking trajectory begins 0.1 units from the surface of the sphere. Consequently, instead of planning the approach all the way to the docking target, the trajectory planner is required to find a trajectory which matches up with the start of the predetermined final approach. A similar technique is used in the other examples as well.

A spherical obstacle with a radius of 0.3 units was placed at (4.0, 2.2, 0.0) and a box–shaped obstacle with sides of length (2.4, 0.4, 1.0) was placed with its center at (1.0, 1.0, 0.5). The obstacle penalty function support interval was 1.0 for each obstacle. The specified completion time for this example is 10 seconds. The vehicle has a mass of one unit; its rotational inertia matrix is the identity matrix. The thrusters are symmetric with maximum force output of 30,000 units; the maximum torque output is 1000 units. The torque gain $\mathbf{R} = 0.1\mathbf{I}$.

The initial translational solution estimate was generated by fitting a cubic polynomial to the required initial and final positions and velocities; the initial and final accelerations were zero. The attitude estimate similarly fitted a cubic to the initial and final modified Rodriguez parameters and their first derivatives. The initial knot points for the spline used by bvp4c were uniformly spaced.

The resulting trajectory is shown in figure 4. This trajectory required a total force of 2.98 units and a total torque of 0.121 units.

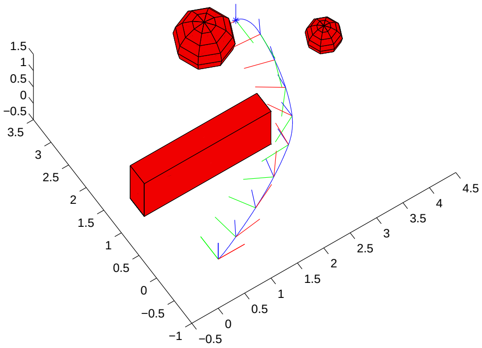


Figure 4: Trajectory plot for example 1 with stationary obstacles. Body axes shown at one second intervals.

5.3 Moving Obstacles

The trajectory planning algorithm is also able to deal with translating and rotating obstacles. To demonstrate this, the spherical obstacle in the previous example was modified to translate and the box–shaped obstacle was modified to translate and rotate. The motion of the obstacles is such that each of them would collide with the spacecraft if it were following the trajectory found in the previous example. Figure 5 shows both the trajectory from the previous example and the new trajectory. As shown, the trajectory planner has modified both the shape of the path and the velocity profile of the spacecraft in order to avoid the obstacles and still dock to the target in the specified time. The force required for this trajectory is 3.03 units; the required torque is 0.133 units.

This example demonstrates that the trajectory planner can successfully find trajectories which fulfill given task constraints, including specified initial and final positions and orientations and specified completion time. These requirements can be met even in the presence of moving obstacles, at the cost of somewhat increased fuel use. Finally, the example shows that the planner can find solve problems which require both attitude and translational maneuvering.

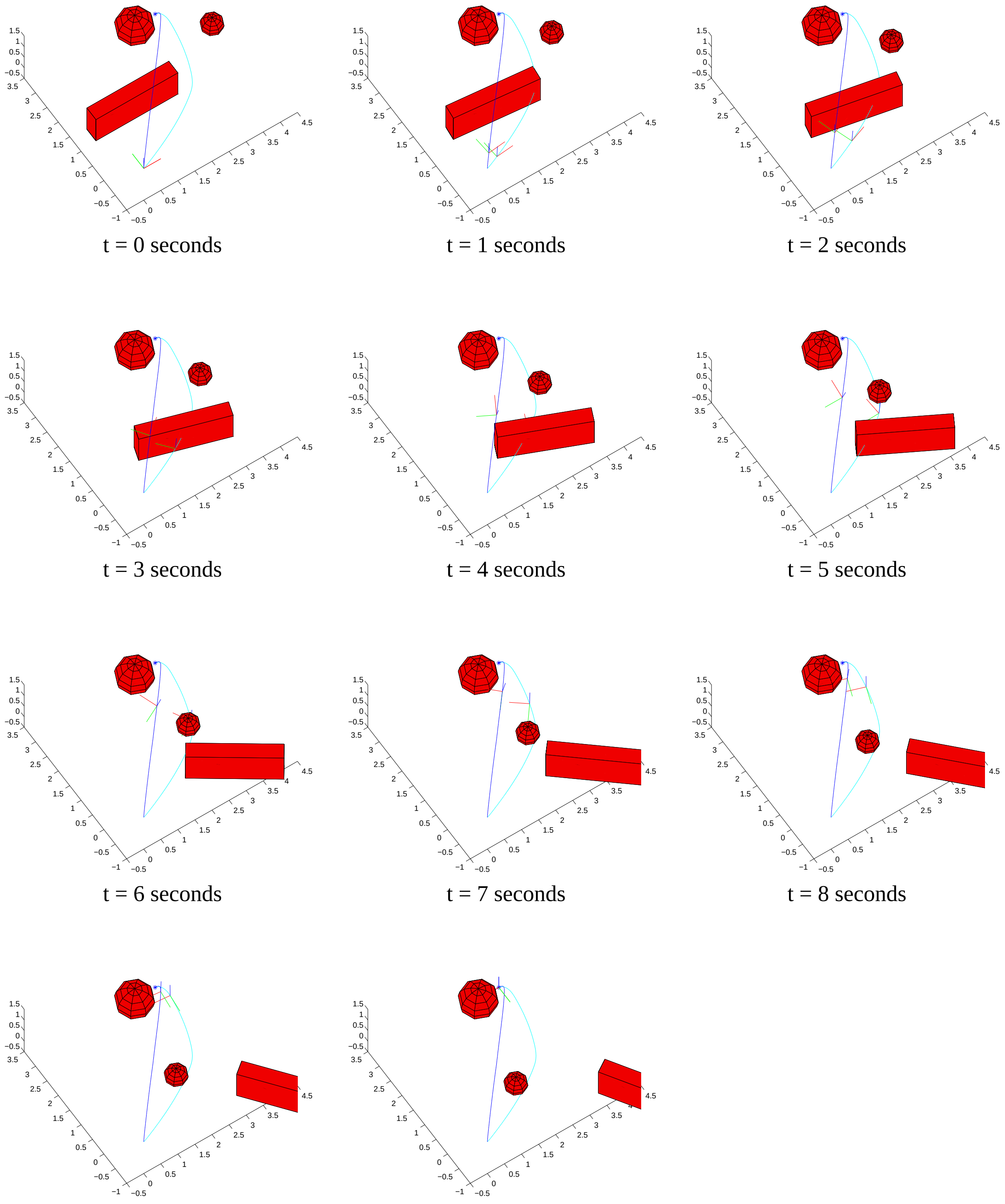
5.4 Orbital Docking

One of the most common methods used to dock the Space Shuttle Orbiter to the International Space Station is via an r–bar approach. During this type of approach the orbiter is required to approach the ISS along the ISS radius vector because along this vector orbital dynamics provide a natural braking force, allowing the approach to proceed without concern for plume impingements.

This example demonstrates such an approach. The ISS is assumed to be in a 388.92 km (210 n.m.) circular orbit with a pitch rate synched to the orbital rate, so that the station maintains a constant orientation with respect to the surface of the Earth. The orbiter begins the approach from the MC4 point, which is located 274 meters (900 feet) below and 549 meters (1800 feet) behind the ISS, with the relative velocity and rotational rates nulled. The orbiter orientation is nose forward, belly down [11]. The orbiter is required to dock to a target located at the midpoint of the ISS main truss. The target is oriented facing the earth. The required final orientation is nose forward, belly down.

Corresponding with NASA protocol, an approach corridor was added as a constraint. The approach corridor consists of two virtual rectangular obstacles which are located in front of and in back of the docking target and extend 200 meters below the truss. At the its opening, the obstacles are approximately 150 meters apart; at the terminal end they are approximately 9 meters apart. Note that because the trajectory planner penalizes obstacle approach distance more highly as the distance becomes smaller, the effective width of the approach corridor is smaller than the actual distance between the two obstacles. The obstacles are each 60 meters wide. The approach corridor is shown in figure 6. The obstacle avoidance gain for the corridor obstacles was set at $\alpha = 1000$. The ISS model is shown in figure 6. The approach corridor is shown in wireframe to emphasize the fact that the approach corridor obstacles are virtual and do not represent physical objects.

For this example the orbiter mass was assumed to be 117,000 kgs. The rotational inertias were $I_{xx} = 1.29 \times 10^6$, $I_{yy} = 9.86 \times 10^6$, and $I_{zz} = 10.1 \times 10^6$ $\mathrm{N\text{–}m/s}^2$. The orbiter main OMS thrusters were assumed to be disabled; this is in



t = 0 seconds

t = 1 seconds

t = 2 seconds

t = 3 seconds

t = 4 seconds

t = 5 seconds

t = 6 seconds

t = 7 seconds

t = 8 seconds

t = 9 seconds

t = 10 seconds

Figure 5: Moving obstacles. Darker trajectory is for moving obstacle case. Lighter trajectory is for nominal, stationary obstacle case, and is presented for comparison.

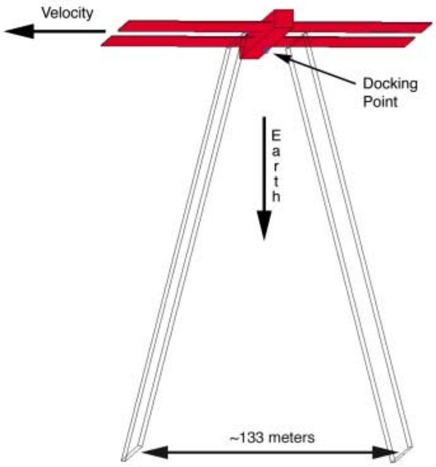


Figure 6: International Space Station model for Example 2, showing approach corridor.

keeping with NASA standard procedure, which stipulates that only the RCS thrusters are to be used for ISS final approach. The shuttle thruster model is slightly unrealistic, however, in that here the thrust was assumed to be symmetric in both the positive and negative body axis directions. RCS engines output 3870 Newtons of thrust each; this example assumes six thrusters aligned along each body axis, for a total thrust output of $\pm 23,220$ Newtons. Also unlike the real orbiter, here the force and torque capabilites are assumed to be decoupled, as if reaction wheels or control–moment gyros were being used to control the shuttle attitude. Here the torque output is assumed to be $\tau_{roll} = \pm 1484$ N–m, $\tau_{pitch} = \pm 7632$ N–m, $\tau_{yaw} = \pm 7632$ N–m.

This planning task is carried out using the relative dynamic formulation given in equation 7. The completion time was free, with an initial estimate of $t_f = 2200$ seconds. The torque gain was 10; the completion time penalty gain γ was 0.5. Here continuation was used on both the obstacle penalty gain and the AME solution accuracy (all but the last continuation iterations operated with a fairly loose solution tolerance; a final pass improved the accuracy). The obstacle support interval was 0.1 km. It was found that the trajectory planner converges to the correct solution much more quickly if the various parts of the state are normalized; therefore, the problem was formulated in the planner using units of kilometers, hours, and orbiter mass units. The data presented here have been converted back to seconds and kilograms, however.

The initial estimate used here was computed by first solving the problem without the approach corridor in place; the continuation schedule used for this computation was $\alpha = (0, 2.5)$; $\epsilon = (0, 0.9, 0.99, 0.995, 0.999)$. For the second step of the solution, with the approach corridor in place, continuation was used on the width of the approach corridor.

The trajectory found by the trajectory planning algorithm is shown in figure 7. Note that this plot is expressed in the ISS body frame, which is rotating with respect to the inertial. It is important to note, however, that since the planning algorithm works in a reference frame which is fixed to the target frame but which is non-rotating, the approach corridor is in fact rotating in pitch at -2π radians every orbital period. Thus the trajectory planner successfully found a collision–free trajectory which traverses a tightly constrained, moving corridor.

Although not a primary concern for orbiter docking maneuvers, the completion time of 2383 seconds (39.71 minutes) compares favorably to the pilot–flown approach, which requires approximately 4020 seconds (67 minutes) from the MC4 point. The total force required for the approach is 138,941 Newton–seconds.

A nominal pilot–flown approach uses 114 kgs of fuel on average. The orbiter RCS engines use a nitrogen tetroxide/monomethyl hydrazine fuel–oxidizer mix, which attains a specific impulse of 260–280 seconds. The pilot–flown approach should therefore require approximately 291,544 Newton–seconds of thrust [11]. The trajectory planner thus compares quite favorably with human–piloted approaches. By way of comparison, a pure impulsive transfer from the orbiter's initial orbit to its required final orbit which ignores the phasing requirements with the ISS requires 72,823 Newton–seconds of thrust.

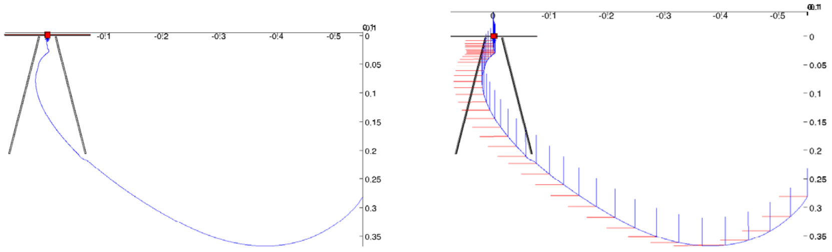


Figure 7: Realistic Space Shuttle Orbiter/International Space Station r–bar approach with approach corridor. Body axes are shown at approximately 60 second intervals. Trajectories are shown with respect to the rotating ISS body frame.

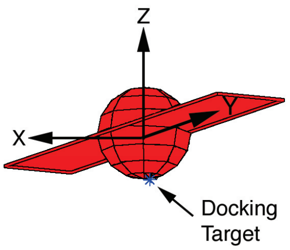


Figure 8: Tumbling satellite model. Satellite is rotating in pitch at 1.14×10^{-3} radians per second and in yaw at 5.2×10^{-2} radians per second. Initial orbital velocity is in the +Z direction.

5.5 Docking to a Tumbling Target

The final example demonstrates a realistic task for a robotic satellite servicer which is beyond the ability of current spacecraft trajectory planners. A satellite which is in the same orbit as the ISS but which is trailing the station by approximately 63 kilometers has malfunctioned and started to tumble. The satellite is modelled as a three meter wide sphere with solar panels extending 7 meters beyond the body of the satellite along the Y axis. The solar panels are three meters high. The docking point on the satellite is located on the main body exactly between the solar panels. The pitch rate of the satellite with respect to the inertial is zero, which implies a pitch rate of approximately 1.14×10^{-3} radians per second (one full rotation every 92 minutes) with respect to the ISS. The satellite also has a yaw rate of approximately 5.2×10^{-2} radians per second (one full rotation every two minutes). The trajectory planner is required to find a trajectory which begins with the servicer docked to a point on the ISS located 10 meters from the center of the ISS truss, facing the Earth, and which ends with the servicer docked to the tumbling satellite. The satellite is shown in figure 8.

The robotic servicing vehicle is assumed to have a mass of 2000 kgs and rotational inertias of $I_{xx} = I_{yy} = I_{zz} = 3200 \text{ N–m/s}^2$. The maximum force output is ±1000 N along each axis, and the maximum torque output is ±2000 N–m. The torque penalty gain was 10 and the time gain γ was 0. As in the previous example, the problem was formulated in nonstandard units: here, kilometers, minutes, and servicer mass units were used. Again, the data have been converted back into standard units for presentation purposes.

The results are shown in figure 9. This trajectory requires 81,611 Newton–seconds of thrust and 66.5 Newton–meter–seconds of torque. The entire maneuver takes 2728 seconds (45.46 minutes). The overall orbital intercept trajectory is shown in figure 9. A time sequence of the terminal phase approach to the satellite is shown in figure 10. Close examination shows that the servicer "spins up" in order to match the yaw rotational rate of the satellite for several minutes prior to docking.

This example demonstrates clearly that the trajectory planner can solve problems which require high levels of coordination between translational and attitude maneuvers. As in the previous example, it can also find trajectories which satisfy realistic bang–coast–bang thruster constraints. Finally, this example shows that the trajectory planner can solve problems which require the spacecraft to move over distances large enough that Hill's equations and other linearized relative orbital equations of motion are not valid. Thus, the trajectory planner can solve problems which are beyond the ability of any previously described orbital trajectory planner.

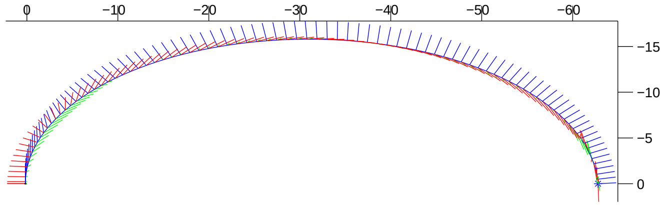
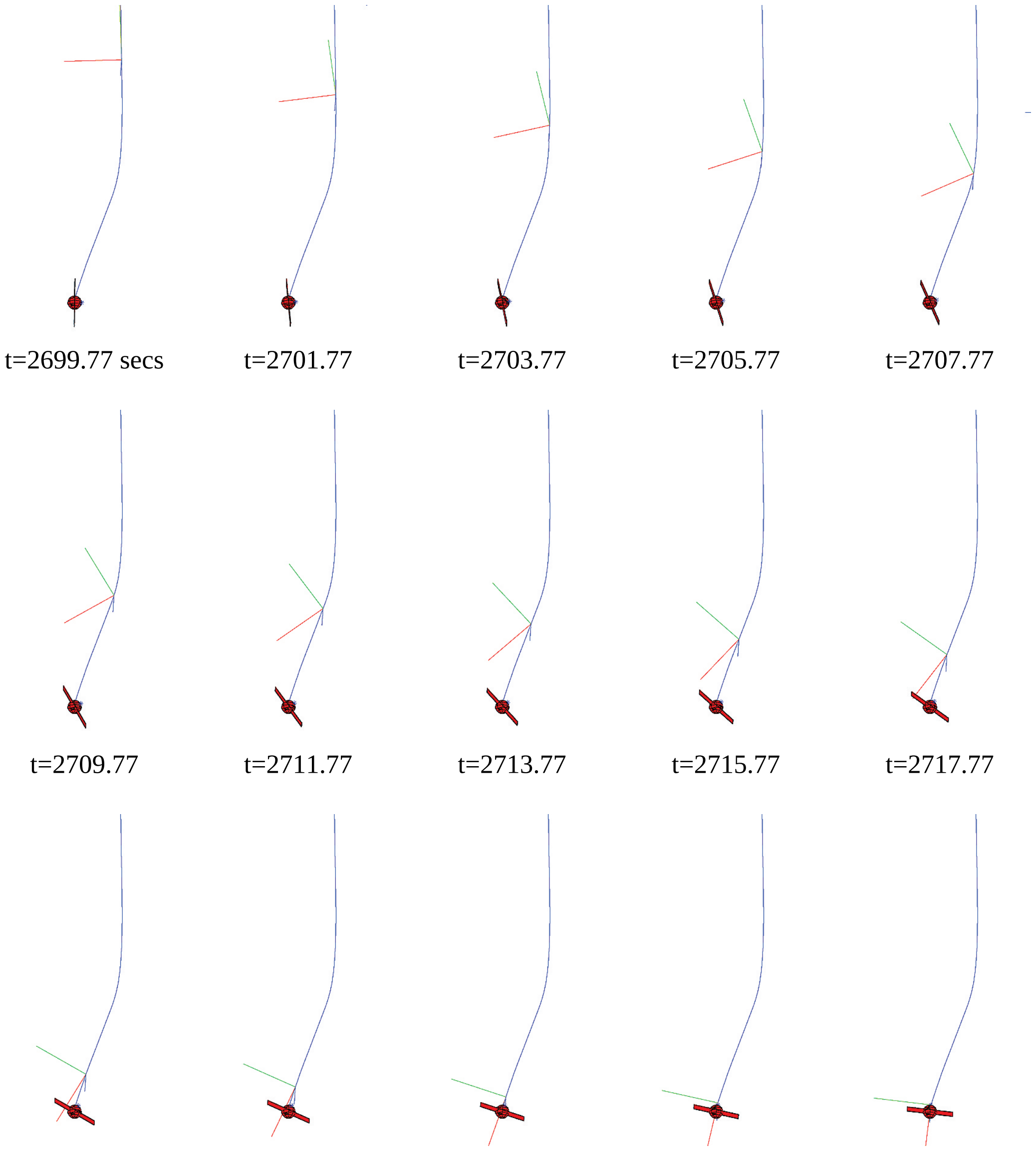


Figure 9: Satellite servicer docking problem trajectory. Body axes are shown at approximately 30 second intervals.



t=2699.77 secs

t=2701.77

t=2703.77

t=2705.77

t=2707.77

t=2709.77

t=2711.77

t=2713.77

t=2715.77

t=2717.77

t=2719.77

t=2721.77

t=2723.77

t=2725.77

t=2727.77

Figure 10: Satellite servicer approach trajectory. Body axes shown every 2 seconds.

6 Conclusion

As the examples demonstrate, the variational trajectory planning algorithm developed in this dissertation fulfills the requirements set forth in the introduction. In particular, the algorithm can

- find free paths in the presence of stationary or moving obstacles
- find minimum–fuel trajectories even in the presence of gravitational fields and other environmental forces
- deal with a wide range of endpoints constraints and time scales
- take into account realistic spacecraft dynamic models
- take into account realistic thrusters, including bang–coast–bang thrusters

Also, notably, the dynamic models used here are not the linearized dynamics used in many spacecraft trajectory planners. As a consequence, the algorithm can find minimum–fuel trajectories over much larger separation distances than most other trajectory planners. Also, the variational trajectory planner is not limited to translational trajectory planning; as the last example demonstrates, the algorithm can solve problems which require tightly coupled translation and rotation. These capabilities, taken as a whole, indicate that the variational trajectory planning algorithm described here is more capable than any spacecraft trajectory planner currently extant in the literature.

References

[1] http://ic.arc.nasa.gov/projects/psa/index.html.

[2] http://www.boeing.com/news/releases/1997/news_release_971015b.html.

[3] http://www.estec.esa.nl/spaceflight/atv.htm.

[4] http://www.darpa.mil/tto/PROGRAMS/astro.html.

[5] Howie Choset, J. J. Fernandez, and David Kortenkamp. Path planning and control for aercam, a free–flying inspection robot in space. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation (ICRA99)*, pages 1396–1403, Detroit, MI, 1999.

[6] Howie Choset, Marco La Civita, and Jong Chul Park. Path planning between two points for a robot experiencing localization error in known and unknown environments. In *Proceedings of the International Conference on Field and Service Robotics*, Pittsburgh, PA, 1999.

[7] W. H. Clohessy and R. S. Wiltshire. Terminal guidance system for satellite rendezvous. *Journal of Aerospace Science*, pages 653–658, 674, September 1980.

[8] Peter Deuflhard. Nonlinear equation solvers in boundary value problem codes. In B. Childs, M. Scott, J. W. Daniel, E. Denman, and P. Nelson, editors, *Codes for Boundary–Value Problems in Ordinary Differential Equations*, volume 76 of *Lecture Notes in Computer Science*, pages 40–66. Springer, 1979.

[9] Leonhard Euler. Methodus inveniendi lineas curvas: maximi minimive proprietate gaudentes sive solutio problematis isperimetrici latissimo sensu accepti. In Constantin Caratheodory, editor, *Opera omnia series I*, volume 24. M. M. Bousquet, 1744.

[10] C.A. Floudas. *Nonlinear and Mixed–Integer Programming — Fundamentals and Applications*. Oxford University Press, 1995.

[11] Jorge Frank. Personal communication with Jorge Frank, Rendezvous/Proximity Operations instructor, Spaceflight Training Division, NASA/Johnson Space Center.

[12] Geza S. Gedeon. Orbital mechanics of satellites. In *Proceedings of the American Astronautical Society*, August 1958.

[13] George W. Hill. On the part of the motion of the lunar perigee which is a function of the mean motions of the sun and moon. *Acta Mathematica*, 8:1–36, 1886.

[14] Rudolf F. Hoelker and Paul S. Silber. The bi–elliptical transfer between coplanar circular orbits. In *Proceedings of the Fourth AFBMD/STL Symposium*, volume 3, pages 164–175, New York, 1961. Pergamon Press.

[15] Walter Hohmann. Die erreichbarkeit der himmelskörper (the attainability of celestial bodies). Munich, 1925.

[16] Mark Charles Jackson. A six degree of freedom, plume–fuel optimal trajectory planner for spacecraft proximity operations using an a* node search. Master's thesis, Massachusetts Institute of Technology, 1994.

[17] O. Khatib. *Commande Dynamique dans l'Espace Opérationnel des Robots Manipulateurs en Présence d'Obstacles*. PhD thesis, Ecole Nationale Supérieure de l'Aéronautique et de l'Espace, Toulouse, 1980.

[18] O. Khatib. Real time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1), Spring 1986.

[19] Jacek Kierzenka and Lawrence F. Shampine. A bvp solver based on residual control and the matlab pse. *ACM Transactions on Mathematical Software (TOMS)*, 27(3):299–316, 2001.

[20] Donald E. Kirk. *Optimal Control Theory: An Introduction*. Prentice–Hall, 1970.

[21] David G. Luenberger. *Optimization by Vector Space Methods*. John Wiley & Sons, New York, 1969.

[22] C. Ó'Dúnlaing and C. K. Yap. A retraction method for planning the motion of a disc. *Journal of Algorithms*, pages 104–111, 1982.

[23] Colin R. McInnes. Autonomous path planning for on–orbit servicing vehicles. In *Reducing Space Mission Cost*. British Interplanetary Society, April 1999.

[24] F. McQuade and C. R. McInnes. Autonomous control for on–orbit assembly using potential function methods. *The Aeronautical Journal*, pages 255–262, June/July 1997.

[25] Muruhan Rathinam. Numerical solution of optimal trajectories for fully actuated mechanical systems. Division of Engineering and Applied Science, California Institute of Technology, Technical Memorandum CDS 99–005, October 1999.

[26] Arthur Richards, Tom Schouwenaars, Jonathan P. How, and Eric Feron. Spacecraft trajectory planning with collision and plume avoidance using mixed–integer linear programming. *Accepted, AIAA Journal of Guidance, Control and Dynamics*, July 2002.

[27] Alexander B. Roger and Colin R. McInnes. Safety constrained free–flyer path planning at the international space station. *Journal of Guidance, Control, and Dynamics*, 23(6):971–979, November–December 2000.

[28] Lawrence F. Shampine, Jacek Kierzenka, and Mark W. Reichelt. Solving boundary value problems for ordinary differential equations in matlab with bvp4c. The MathWorks, Inc. tutorial, 2000.

[29] Ronald J. Simmons, Edward V. Bergmann, Bruce A. Persson, and Walter M. Hollister. Six dimensional trajectory solver for autonomous proximity operations. In *Proceedings of the AIAA Guidance, Navigation and Control Conference*, pages 1291–1303. American Institute of Aeronautics and Astronautics, 1990. AIAA paper 90–3459.

[30] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer–Verlag, New York, 1993.

[31] Wayne Tempelman. Minimum–energy intercepts originating from a circular orbit. *Journal of the Aerospace Sciences*, 28:924–929, December 1961.

[32] Panagiotis Tsiotras. Stabilization and optimality results for the attitude control problem. *AIAA Journal of Guidance, Control, and Dynamics*, 19(4):772–779, July–August 1996.